**Overview**

This program is written to establish tcp and udp client /server paradigm for a DVD store ordering system. The client can send two types of requests to server, LIST and ORDER. For LIST request, server will reply with the list of items available in inventory and for ORDER request, client will reply with a message to indicate whether the order was successfully processed or not.

**List of Files:**

ex1\_client.c: The program for tcp client

ex1\_server.c: The program for tcp server

ex2\_client.c: The program for udp client

ex2\_server.c: The program for udp server

Makefile: A makefile for compiling all the 4 programs

**ex1\_client.c:**

This program contains the client algorithm to send list and order request to server over tcp connection.

1. The user request is set in request buffer and connectTCP() method is called to establish connection
2. int connectTCP (const char \*addr, const char \*port): Set the protocol port number and IP address of the server with which communication is desired, allocate a socket, connect the socket to the server and returns the socket descriptor.

**Server IP: 127.0.0.1, port: 8888**

1. Send the user request to the server. Then processlistdata () or processorderdata() method is called to process the inventory list or order status reply from server.
2. void processlistdata(int csd): Receive the response sent by the server for the list request and display the list of items in a tabular format in client console.

void processorderdata(int csd,char resBuffer[]) : Receive the response sent by the server for the order request and the reply message is printed on the client console.

1. Close the socket
2. Exit the program

int handleerror(const char \*errmsg): Handles the error that can occur while establishing the connection with server. It prints the error message in console and exits the program.

**ex1\_server.c:**

This program contains the server algorithm to process the requests sent by the client and send the reply back to client.

1. loadInventoryData() method is called which populates the inventory data.
2. passiveTCP() is called which will create the socket , bind to the well known address ,set the server in passive mode and returns the socket descriptor.

**Server IP: 127.0.0.1 , port: 8888**

1. Accept the connection request from client and slave socket is created.
2. Receive the request. Based on the request type from client , processlistrequest() or processorderrequest () method is called.
3. processlistrequest(int ssd): Send the inventory list back to client.

void processorderrequest(int ssd,char reqBuffer[],char resBuffer[]): Checks if the requested item is present in inventory. If yes , check if enough quantity is available . If yes , send ok response to client. If enough quantity is not available or the item is not available in inventory , appropriate messages will be sent.

1. Close the slave socket.

Go back to step 3 to accept next connection request.

int handleerror(const char \*errmsg): Incase of error , displays the error message and exits the program.

**ex2\_client.c:**

This program contains the client algorithm to send list and order request to server over udp connection.

1. The user request is set in request buffer and connectUDP() method is called.
2. int connectUDP (const char \*addr,const char \*port): Set the protocol port number and IP address of the server with which communication is desired , allocate a socket and returns the socket descriptor.

**Server IP : 127.0.0.1 , port : 8888**

1. Send the user request to the server. Then processlistdata () or processorderdata () method is called to process the inventory list or order status reply from server.
2. void processlistdata(int csd): Receive the response sent by the server for the list request and display the list of items in a tabular format in client console.

void processorderdata(int csd,char resBuffer[]) : Receive the response sent by the server for the order request and the reply message is printed on the client console.

1. Close the socket
2. Exit the program

int handleerror(const char \*errmsg): Handles the error that can occur while establishing the connection with server. It prints the error message in console and exits the program.

**ex2\_server.c:**

This program contains the server algorithm to process the requests sent by the client and send the reply back to client.

1. loadInventoryData() method is called which populates the inventory data.
2. passiveUDP() is called which will create the socket , and bind to the well known address.

**Server IP: 127.0.0.1, port: 8888**

1. Receive the request. Based on the request type from client, processlistrequest () or processorderrequest () method is called.
2. processlistrequest (int ssd): Send the inventory list back to client.

void processorderrequest (int ssd, char reqBuffer [], char resBuffer []): Checks if the requested item is present in inventory. If yes, check if enough quantity is available. If yes, send ok response to client. If enough quantity is not available or the item is not available in inventory, appropriate messages will be sent.

Go back to step 3 to receive next request.

int handleerror (const char \*errmsg): Incase of error, displays the error message and exits the program.

**Makefile**: This file will compile all the 4 program files and generate the executables ex1\_client, ex1\_server, ex2\_client and ex2\_server.

**Compilation and Execution of programs:**

**TCP**: Run “make” command to execute the Makefile which will compile the client and server program.

Start the server program in one terminal window (./ex1\_server)

Start the client program in another terminal (./ex1\_client along with the command line inputs , e.g.: ./ex1\_client 127.0.0.1 8888 list)

**UDP** : Run “make” command to execute the Makefile which will compile the client and server program.

Start the server program in one terminal window (./ex2\_server)

Start the client program in another terminal (./ex2\_client along with the command line inputs , e.g.: ./ex2\_client 127.0.0.1 8888 list)